Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Информатика»

Тема: Двунаправленные списки

Семестр 2

Выполнил работу

Студент группы ИВТ-22-2б

Ищенко Дарья Олеговна

Проверил

Доцент кафедры ИТАС

Полякова О. A.

Г. Пермь-2022

**Постановка задачи**

Двунаправленный (двусвязный) список – это структура данных, состоящая из последовательности элементов, каждый из которых содержит информационную часть и два указателя на соседние элементы. При этом два соседних элемента должны содержать взаимные ссылки друг на друга.

В таком списке каждый элемент (кроме первого и последнего) связан с предыдущим и следующим за ним элементами. Каждый элемент двунаправленного списка имеет два поля с указателями: одно поле содержит ссылку на следующий элемент, другое поле – ссылку на предыдущий элемент и третье поле – информационное. Наличие ссылок на следующее звено и на предыдущее позволяет двигаться по списку от каждого звена в любом направлении: от звена к концу списка или от звена к началу списка, поэтому такой список называют двунаправленным.

Двунаправленный список отличается от однонаправленного тем, что у каждого элемента существует связь с предыдущим. Кроме того, двунаправленный список хранит указатель не только на первый элемент, но и на самый последний.

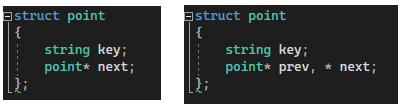


Рис. 1 - Сравнение структур однонаправленного и двунаправленного списка соответственно

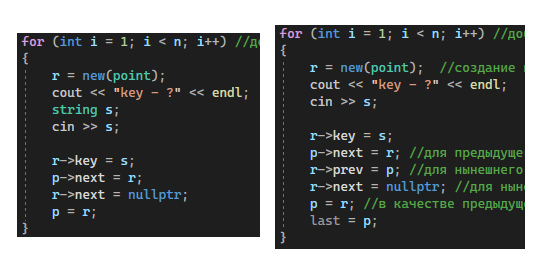


Рис. 2 - Сравнение циклов в функции создания списков для однонаправленного и двунаправленного списка соответственно

**Алгоритм решения**

Создать структуру struct point, которая будет иметь в качестве полей ключевое слово типа string, и указатели на предыдущую и следующую структуру. В функции будут подаваться ссылки на первый и последний элемент двунаправленного списка.

Функция void make\_list(point\*&first, point\*&last), которая создаёт список, связывая узлы друг с другом. Сначала задаётся первый элемент, который имеет два нулевых указателя. При добавлении нового элемента указатель нынешнего узла на следующий элемент (next) переприсваивается на добавляемый элемент. У нового элемента указатель на предыдущий элемент (prev) указывает на нынешний элемент, а next является нулевым.

Функция void print\_list(point\* beg) выводит ключевые значения элементов списка.

Функция void del\_point(point\* &beg, int k, point\*&fin) удаляет элемент по номеру. В ней для предыдущего для k элемента значение указателя next переобозначается на следующий после k элемент. Аналогично, для следующего для k элемента значение prev переобозначается на предыдущий для k элемент. После этого указатель на k элемент удаляется.

Функция void add\_point(point\*& beg, int k, point\*&fin) добавляет элемент в список. Сначала находится k-1 элемент, и его значение next переобозначается на новый элемент

Функция int find\_key\_index(point\* beg) проходит по всему списку и сравнивает значение каждого элемента с введённым словом. Если в списке есть элемент с таким ключом, то выводит номер его позиции.

Функция void delete\_list(point\*& beg, point\*&fin) считает количество элементов в списке и удаляет их с конца.

Функция void printINfile(point\*beg, string path) построчно записывает ключевые значения элементов списка, начиная с первого в файл.

Функция void readFROMfile(string path, point\*&first, point\*&last) имеет одинаковый принцип действия с make\_list(). Функция создаёт список, ключевые значения элементов которого построчно достаются из файла.

Функция void menu(const int c) для управления списком и вызова функций.

**Блок-схема**

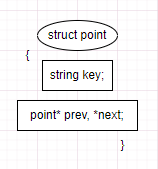
****

Рис. 3 - Блок-схема перегруженных функций ввода и вывода массива

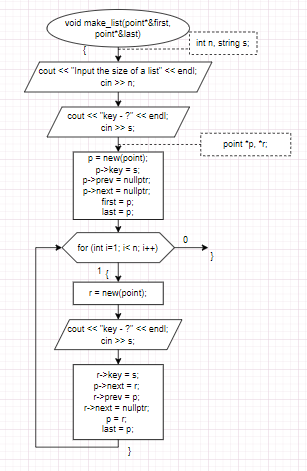
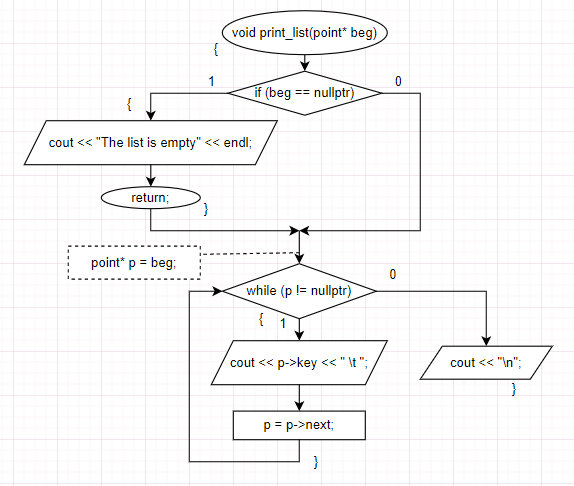


Рис. 4 - Блок-схема функции make\_list(), которая создает двунаправленный список

  
Рис. 5 - Блок-схема функции print\_list(), которая выводит список в консоль

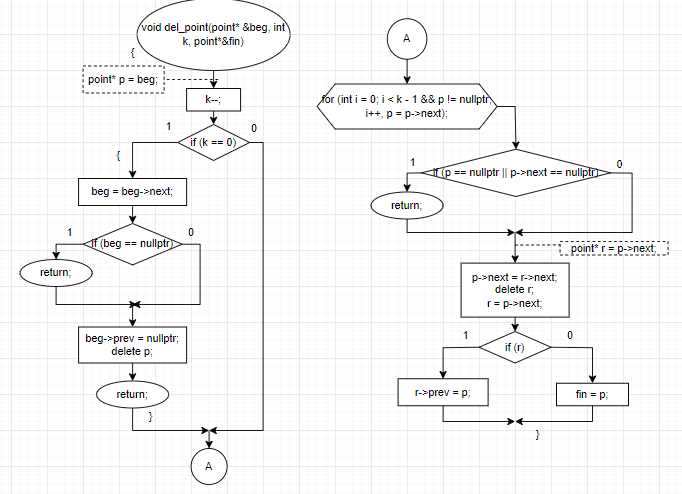


Рис. 6 - Блок-схема функции del\_point(), которая удаляет элемент из списка

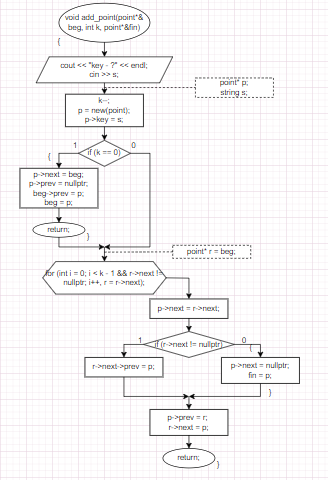


Рис. 7 - Блок-схема функции add\_point(), которая добавляет в список элемент

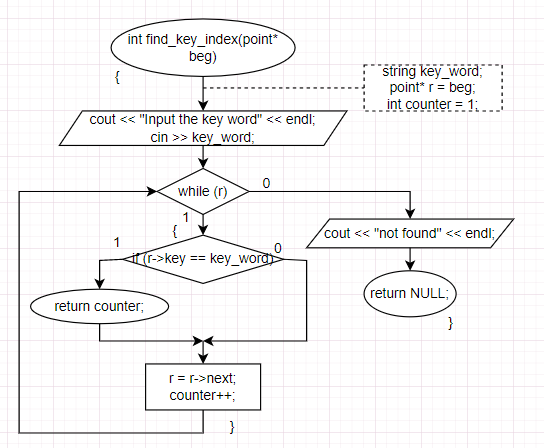


Рис. 8 - Блок-схема функции find\_key\_index()

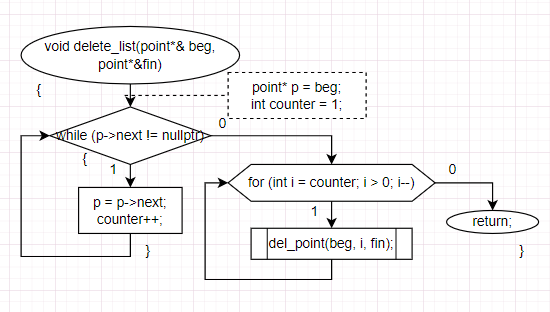


Рис. 9 - Блок-схема функции delete\_list()

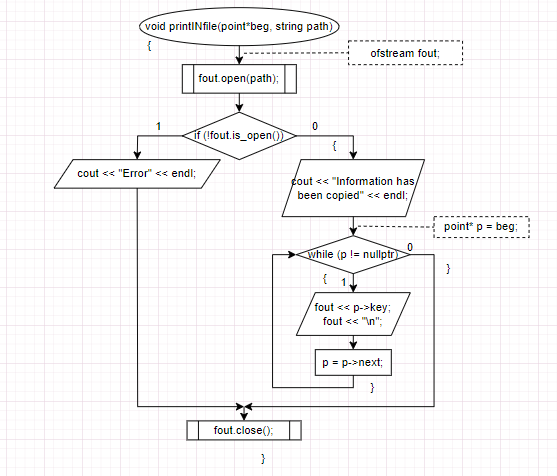


Рис. 10 - Блок-схема функции printINfile()

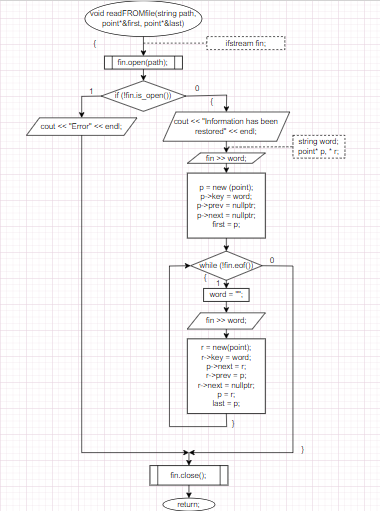


Рис. 11 - Блок-схема функции readFROMfile()

**Текст программы**

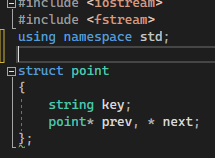


Рис. 12 - Код структуры point на языке С++

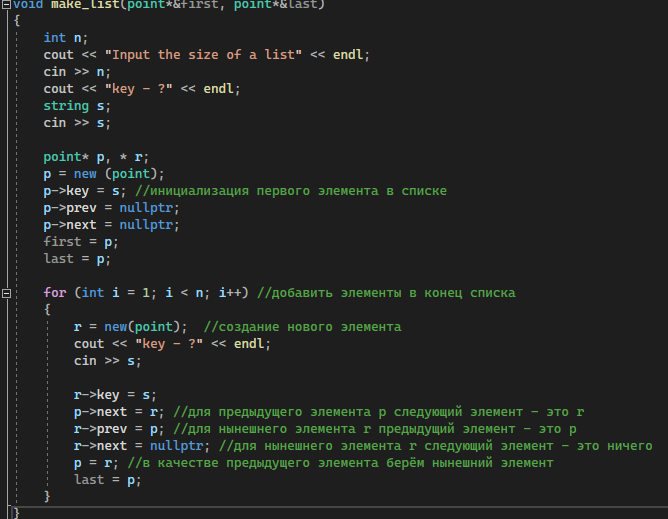


Рис. 13 - Код функции создания списка на языке С++

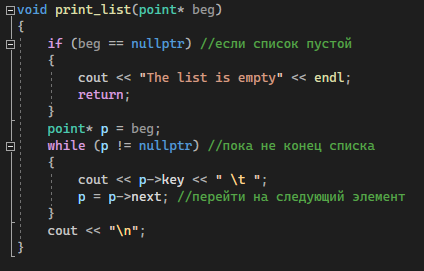


Рис. 14 - Код функции вывода списка в консоль на языке С++

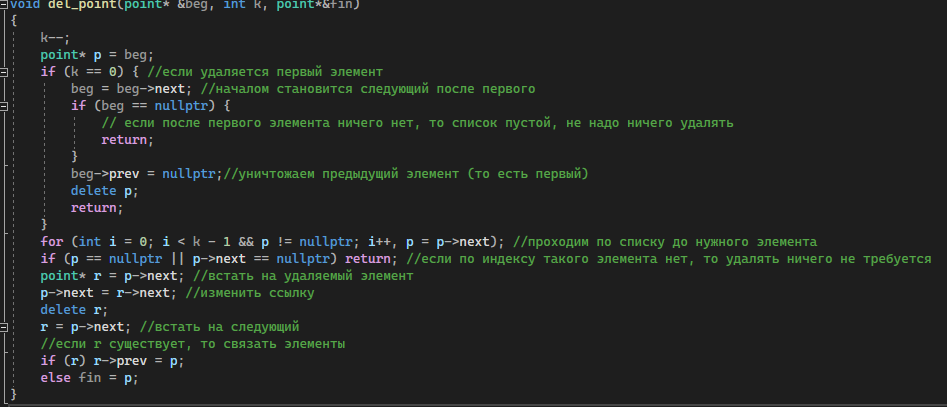


Рис. 15 - Код функции удаления элемента списка на языке С++

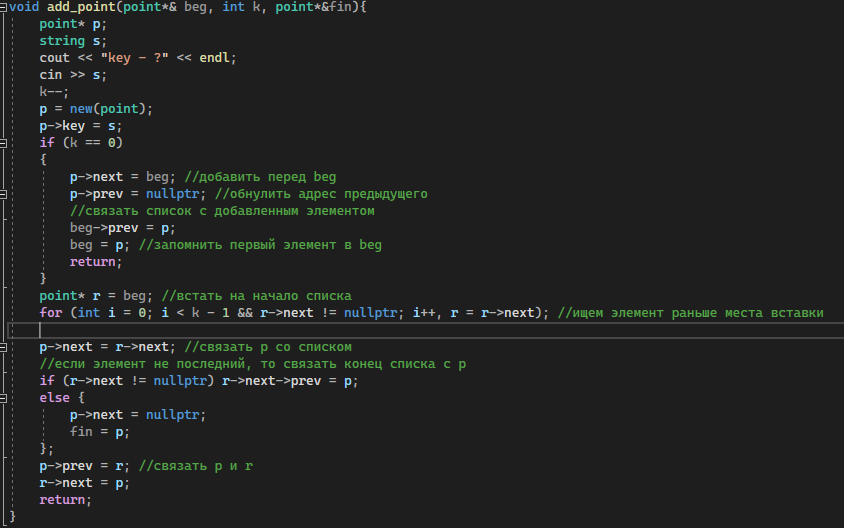
****

Рис. 16 - Функция добавления элемента в список по номеру на языке С++

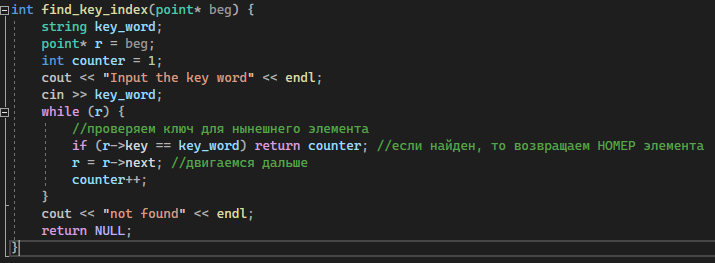


Рис. 15 - Функция поиска номера элемента с заданным ключевым словом на языке С++

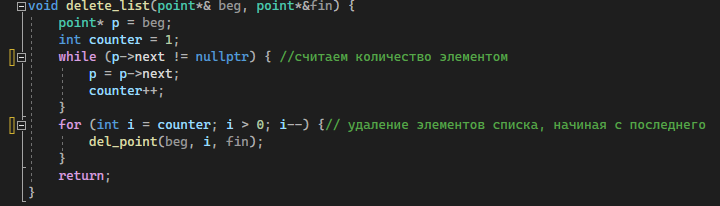


Рис. 16 - Функция удаления списка на языке С++

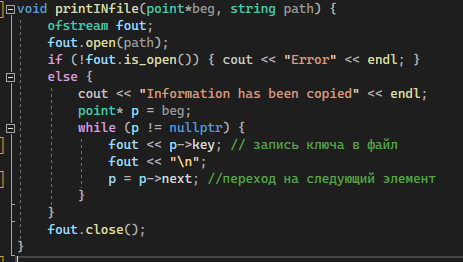


Рис. 17 - Функция записи списка в файл на языке С++

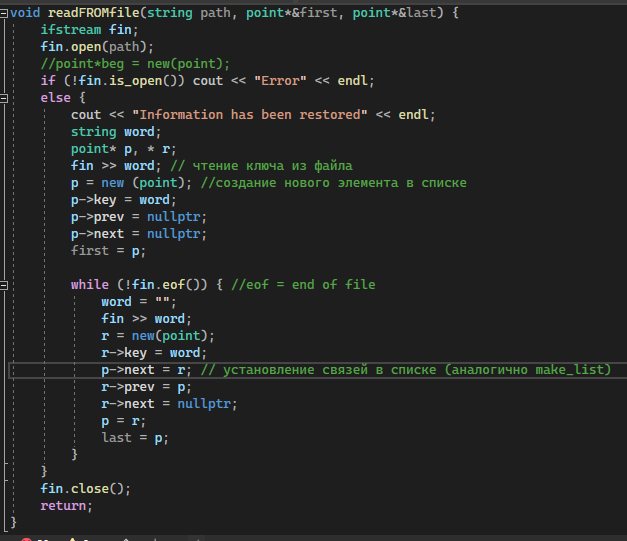


Рис. 18 - Функция чтения списка из файла на языке С++